

# A generic model for Immersive Documentation Environment and applications

Samory KAKEZ<sup>1</sup>, Jean FIGUE<sup>2</sup>, Vania CONAN<sup>3</sup>

**Abstract :** We propose a generic model for designing systems relying on augmented reality techniques in the context of Immersive Documentation Environment (IDE). This model encompasses user/system interaction paradigm, system architecture and exploitation scenario. We illustrate the use of this model on several virtually documented environment systems providing the user with enhanced interaction capabilities. These systems are dedicated to several applications where the operator needs a natural (hands free) access to information, to carry out measurements and/or operate on a devices (e.g. maintenance, instruction). These systems merge live images acquired by a video camera with synthetic data (multimedia documents including CAD models and text) and present the result properly registered in the real world. Vocal commands as well as multimodal interaction, associating speech and gesture, are used to improve interaction.

**Keywords :** Augmented Reality, Human-Computer Interaction, Advanced maintenance systems, Tele-operation, Distributed teaching systems, Mine clearance systems.

## 1. Introduction

Evolution of computer related technologies leads to imagine, in a near future, aided documentation systems providing the user with high level of interaction in his everyday life. *“The need of those people regarding a system to help them in their task can be summarised in the following way : facilities for accessing large databases, efficient information environment (e.g. providing a useful visualisation interface and interaction capabilities) and maybe intelligence in the overall system behaviour (e.g. decision support)”* (Kakez, Conan et al. 1997).

The objective is to provide the user with a simple and efficient tool allowing a task-oriented access to information, without affecting the feeling of immersion into a real environment. Such a tool is called an Immersive Documentation Environment (IDE). It recurses to Augmented Reality (AR) and multimodal interaction to provide the user with task-oriented information registered to his physical environment, in applications where manipulation of information needs to be natural and friendly.

The early research in the field of Augmented Reality (AR) based information visualisation were achieved by Ian Sutherland (Sutherland 1968), who implemented a mixed real and virtual environment based on Bell’s head tracking HMD. This system allows the user to see basic wire frame features superimposed over the vision of the real world. Since the early 1990, many researchers have focused on this area : University of Columbia (Feiner, MacIntyre et al. 1993) and University of North Carolina at Chapel Hill (State, Livingstone et al. 1996) are exploring the concept of AR dedicated respectively to maintenance of manufactured equipment and to surgical assistance.

---

<sup>1</sup> Thomson-CSF LCR, Domaine de Corbeville, 91404 Orsay Cedex France. skakez@worldnet.fr

<sup>2</sup> 1 allée des Tamaris 94480 Ablon sur Seine France, figue@magic.fr (works for Thomson-CSF Optronique),

<sup>3</sup> Thomson-CSF TCS, conan@thomson-lcr.fr

Beside those research activities, industrials manage to set up compact information systems which can be carried on site (Intervisions 1998; Rockwell 1998; ViA 1998; Xybernaut 1998). The concept of wearable computers coins a light personal computer coupled with high level interaction capabilities (Chinnock, Calkins et al. 1996). User can easily access to information (e.g. speech based interaction) that is rendered in his environment (e.g. by the use of low-end CRT based see-through goggles). Nowadays, such wearable systems still do not provide user with a satisfactory immersion sensation. This is essentially due to a lack of correlation between synthetic data and the real world. Many researchers focused on studying the relation between the user (and the physical world), on the one hand, and information, on the other hand. This led Nagao (Nagao and Rekimoto 1995; Nagao and Rekimoto 1996; Rekimoto 1996) to explore possible use of agents in the context of augmented reality : the user interacts with the real (physical) environment through software agents (Maes 1994; Wood, Richardson et al. 1997). Starner (Starner, Mann et al. 1997) has introduced the concept of augmented memory and remembrance agents which outlines the potentiality of augmented reality in terms of perceptual enhancement in an operational context. Those approaches contribute mostly to reduce the gap between heterogeneous concepts (and related technologies). The ultimate objective is not the implementation of a specific method (e.g. registration algorithm or indexation-based retrieval procedure) but rather establishing a generic model depicting an overall behaviour.

Our contribution, through this paper, consists in identifying the basic components of an Immersive Documentation Environment (IDE) and to present an overall organisation of its components in a formal manner. The validity of such a model is illustrated through several applications depicting possible actions of the user in an operational context.

The paper decomposes into four sections. Section 1 introduces the paper. Section 2 presents the proposed IDE model. This model provides a general framework for the design of operational IDE applications. The presentation deals with user/system interaction paradigm, system' architecture and operation scenario. Section 3 presents three different examples of applications based on the proposed model. Finally, section 4 concludes the paper.

## **2. A new generic model for IDE**

The purpose of an Immersive Documentation Environment (IDE) is to enable task-oriented access to information, to render this information inside the physical environment of the user, and to allow the user to manipulate this information in a friendly and natural way.

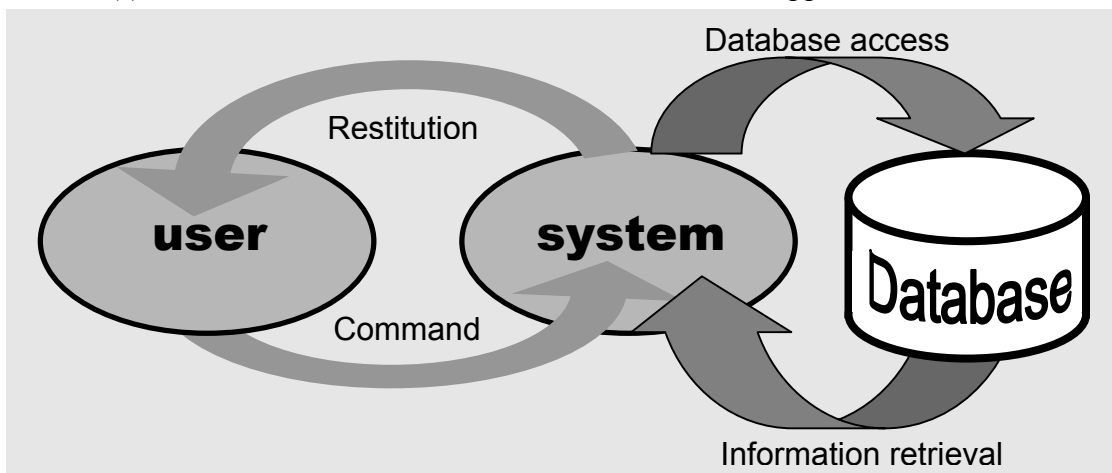
IDE system design is a complicated task since it recourses to varied and complicated technology. Therefore a solid formalism is required to guide the implementation of operational applications. In this section we present a model providing such a formalism. Our objective is to identify the main components of an IDE and to elaborate a descriptive approach of inter-components relations. The presentation stresses on the formalisation of the user/system interaction, the modular decomposition and the relational modelling of the system, and its exploitation scenario. Finally, validity of the proposed model is discussed.

### **2.1. User/system interaction paradigm**

IDE target applications where the user needs a high level of interaction with his system, in a friendly way. Therefore, it is necessary to define precisely the interaction paradigm used in such applications. Figure 1 represents the overall interaction situation in an IDE.

The *user* accesses information by issuing *commands*. These commands can *a priori* take any form : speech, gestures, mouse, joystick etc... User's commands are interpreted by the system and transformed into an appropriate *restitution of information*. The latter consists in redirecting task-related information inside the user's perceptual environment. For this purpose, a *perceptual awareness* mechanism is necessary to enable the system to "perceive" the user's environment. Depending on applications perceptual awareness mechanism can be more or less complicated. In the case of our work it is based on vision technology.

Commands and information restitution thus allow the system to interact with the user inside his perceptual environment. This is an important aspect of IDE. Indeed, such systems are required when the user needs friendly access to information allowing *e.g.* free hands operations and augmented real environment by synthetic documents registered to the physical world. This interaction paradigm can lead to complicated situations implying important amount of knowledge. Therefore, the system is adjunct database(s) to store all relevant information dedicated to the application context.



**Figure 1 :** *user/system interaction paradigm.*

Restitution of information in the user's environment is produced by a set of *actions* of different nature. Actions can be generated by the user (*user generated actions*) or left to the system' initiative (*system generated actions*). In each case, we distinguish several types of actions depending on their complexity. These different types are presented in the following.

***Users generated actions***

These actions are generated by a user (explicit) command. They decompose into the four following types, characterised by the fact that they require or not database access and/or the recourse to perceptual awareness mechanisms.

Action name	access to database	Recourse to perceptual awareness mechanisms
user action 1	no	no
user action 2	yes	no
user action 3	no	yes
user action 4	yes	yes

Action name	Action type	Action sequencing
user action 1	allows the user to send a low-level command to the synthetic environment.	Command + Restitution
user action 2	no influence of physical environment on information retrieved from database, nor on the way it is rendered.	Command + Information Retrieval + Restitution
user action 3	Low level commands requiring perceptual awareness	Command + Perception + Restitution
user action 4	requires perceptual information regarding the user's environment	Command + Information Retrieval + Perception + Restitution

The following examples illustrate these different types of user generated actions.

Action name	User command	System response
user action 1	" <i>Iconify all windows</i> "	All information windows located in the main display window are iconified.
user action 1	" <i>Wire-frame rendering</i> "	3D models located in the synthetic environment are rendered in wire-frame mode.
user action 2	" <i>How to plug the PC to the mains ?</i> "	A schema is displayed in one corner of the display. An audio soundtrack is played.
user action 3	" <i>Iconify this window</i> " - the user makes use of appropriate device (e.g. data glove) in order to achieve a spatial designation -	Designated window is iconified.
user action 4	" <i>How to remove the engine' bonnet?</i> "	An information bubble with the arrow pointing to the bonnet blocking system is registered to that component in the real scene.
user action 4	" <i>How should I remove this ?</i> "	An information bubble with the arrow pointing to the designated object is registered to it.

### ***System generated actions***

These actions are generated using the perceptual awareness of the system to produce a restitution of information. They split into the following types.

- ***system action 1*** : do not require a database access and decompose into :

Perception + Restitution

Example : the most common example is tracking of a real object in the scene. Any modification of the geometry of real scene (including user movements) is detected by the perceptual awareness module and transmitted to the synthetic environment manager to be displayed.

- ***system action 2*** : require a database access and decompose into the sequence :

Perception + Information Retrieval + Restitution

Example : when the user moves in the real environment, physical objects enter or leave his field of view. In the first case, the system identifies these objects and

searches for related information to be displayed in the synthetic environment. In the second case, the system can *e.g.* iconify all information related to this object.

## 2.2. IDE modular breakdown

In Figure 2 we propose a modular IDE decomposition breaking down into five modules.

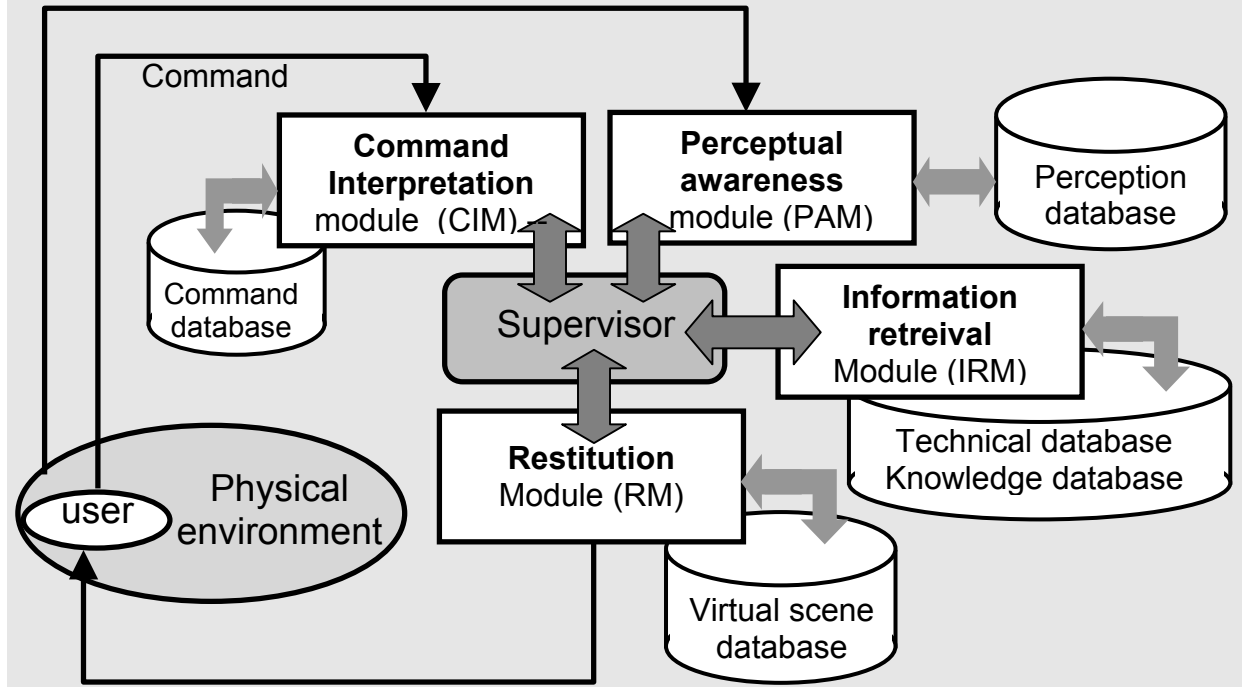


Figure 2 : IDE modular breakdown.

**Command interpretation** and **perceptual awareness** modules (resp. CIM & PAM) constitute the system' inputs. Command interpretation module processes data coming from the user while perceptual awareness module interprets data coming from the physical environment of the user. Perceptual awareness module provides the system with the capability to analyse what happens in the user's environment. It is the homologous to the perceptual system of the user. It can rely on several senses such as vision and audition and can as well make use of other signals beyond the reach of humans (e.g. radar, X rays, ...), depending on applications.

**Restitution** module (RM) constitutes the output of the system. It redirects data to the physical environment of the user. **Information retrieval** module (IRM) manages the access to task-related information. This module is particularly dedicated to management of application technical information and of operation *scenario* (see section 2.4 "The application scenario" for more details). **Supervisor** module has two functions. First it manages possible conflicts between different modules. Moreover, it centralises the application control/command and thus allows to facilitate system maintenance.

In this model, each module is adjunct a dedicated database. This approach makes the system' overall performances better in terms of reaction times and turns out to be particularly necessary when the application becomes sophisticated.

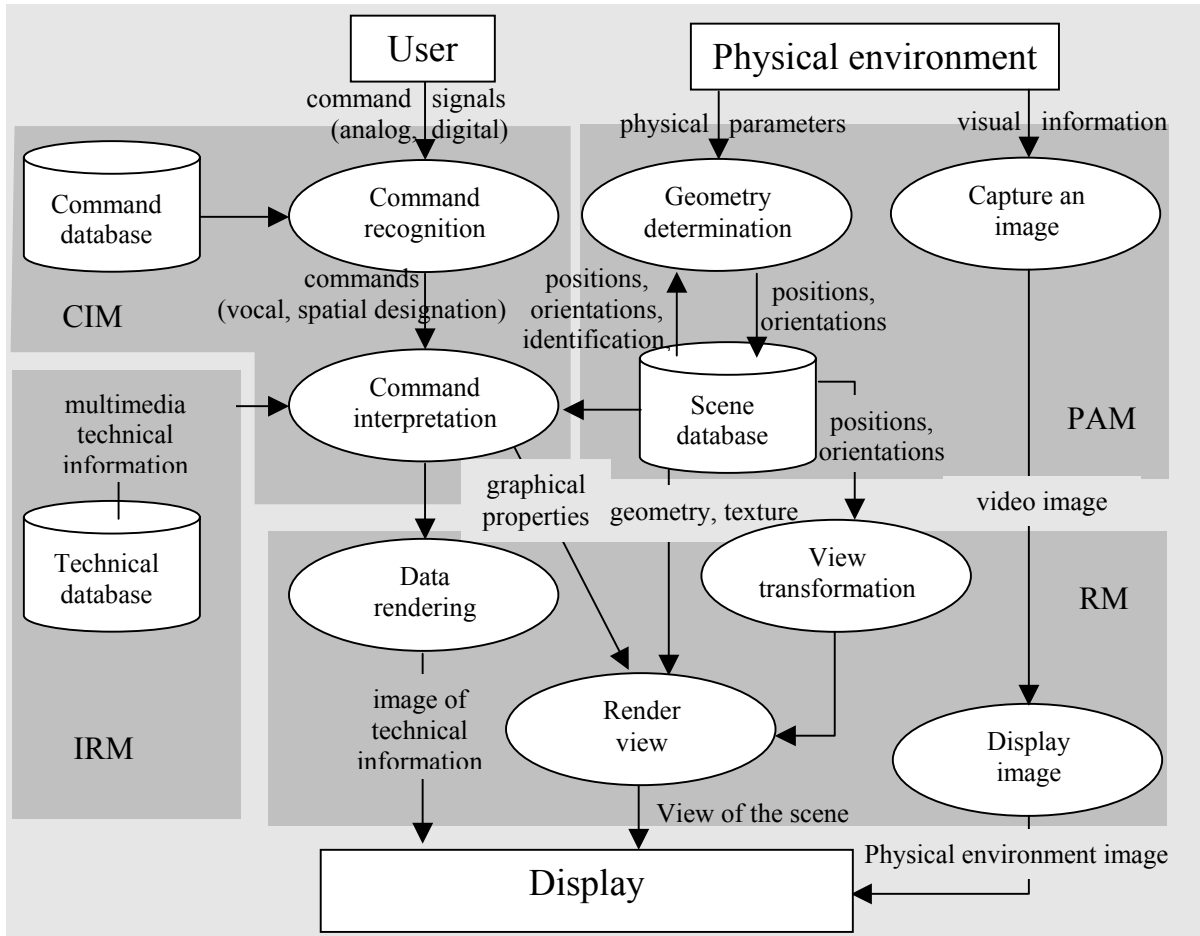
## 2.3. Relational decomposition of the model

Figure 3 illustrates the relational description of the IDE model. This decomposition describes the overall inner data flow. Elements of the IDE modular breakdown (fig. 2) have been recalled by shaded polygons.

Relations are organised around three actors. The *user* and user's *physical environment* constitute the inputs of the flow. The *display* is the output of the flow. Depending on the

application, this element can be e.g. a regular computer screen, video projectors, an HMD or a see through HMD.

Data containers of the modular breakdown (fig. 2) have been included in their corresponding module boxes on fig. 3. The command database contains elements for the command recognition system to decrypt user's commands (e.g. natural language requests, gestures, etc). The technical database contains task-related technical information (procedures, specifications, scenario, etc.). The virtual scene database contains the description of the synthetic environment (camera parameters, objects geometry, colours and textures, etc.). The synthetic environment composes of augmentation elements that will be superimposed to the real scene.



**Figure 3 :** relational model of IDE.

The supervisor is not represented in this figure for clarity. This module manages the data flow symbolised by the arrows in fig. 3.

#### 2.4. The application scenario

The previous model description constitutes the “mechanical” part of an IDE. However, to implement a functional application out of it, we need the functioning conditions of the application. This set of conditions is referred to as the (operation) *scenario*. It describes the way the application will be used practically. For instance, in the case of an application for PC maintenance (such as that described in section 3.1 “Virdoc”), the scenario describes how the application allows diagnostic and maintenance tasks. It contains the list of all possible actions. It also describes how these actions are achieved (e.g. superimpose an information bubble registered to the graphics board in the real scene in answer to the user's question : ”Procedure for graphics board removal”).

As a matter of fact, the operation scenario is the most delicate task involved when designing an IDE for practical applications. Indeed, all relevant tasks required for the purpose of the application must be covered and given a specific restitution according to customers specifications. Therefore, this work is achieved in two steps. The design step consists in tracking all specification requirements in order to provide them the desired answer. In this phase, requirements must be percolated from higher level descriptions (“*I want to be able to see the engine of the car without opening the bonnet*”) down to the finest implementation details (constraints on such or such unix socket). Since this work is tedious it is preferable to recourse to dedicated system architecture design methodologies, such as MIST for instance, to insure respect of specifications and consistency of requirements. This phase being achieved, the system is implemented with the prepared scenario. Then a validation step is performed by extensive testing implying end users. This iteration is generally necessary to fix final presentation details.

### **2.5. Validity of the model**

The generic model presented above provides a recyclable framework for augmented reality and immersive documentation environment applications. However, this presentation hides the difficulty of design of separate modules. In particular, modules linked to perception (command recognition/identification, perceptual awareness) are not easy to tackle, in general. Moreover, information access is also critical in this kind of application, where real time conditions are necessary. Therefore, database management must be handled with great care in the most demanding applications. From our experience a model with centralised control/command (the supervisor) makes applications much more easy to maintain and to make evolve. This approach also simplifies the co-opetition conditions between modules.

This model has been the basis for several pilot applications at Thomson-CSF Corporate Research Labs (see the next section). Therefore we think this approach is particularly well suited to maintenance and tele-operation applications implemented as immersive documentation environments.

## **3. Examples of application of the model**

We describe three applications relying on the model described above. The first one illustrates the use of the model to implement an IDE for PC computers maintenance. This is the more detailed illustration. In this example, user accesses a database containing common maintenance procedures, caution instructions and technical specifications.

The second application concerns a remote teaching/learning system, based on tele-operation. Its basic idea is to share an information environment between the manufacturer of a specific industrial equipment (car bench) and a customer (car repairer). The information environment allows the manufacturer to explain specific procedures to the customer, in a natural manner, and to check that he applies them properly.

The last application illustrates implementation of an IDE for mine clearance instruction. The interface allows users to access mine identification databases and to learn mine neutralization procedures.

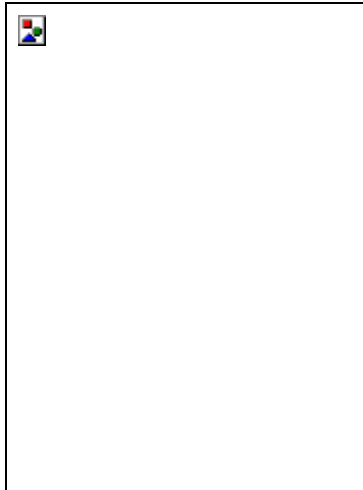
### **3.1. Virdoc**

This application was designed to help computer maintenance personnel identify failures of PC’s central unit. The system is operated through a multi-modal interface (speech

and hand designation commands). The following paragraph illustrates some of the system functionalities by describing user commands and system responses.

The intervention is separated in two steps : diagnostic and maintenance. Diagnostic step consists in identifying symptoms related to the failure. User observes the equipment (PC central unit) without opening it and achieves basic controls on its overall behaviour. Diagnostic step leads to a more detailed inspection involving measurements and testing procedures prior to achieving maintenance.

To begin with, the operator sits next to the damaged PC. He is wearing an HMD and is equipped with a 3D joystick and a microphone as illustrated in fig. 4.

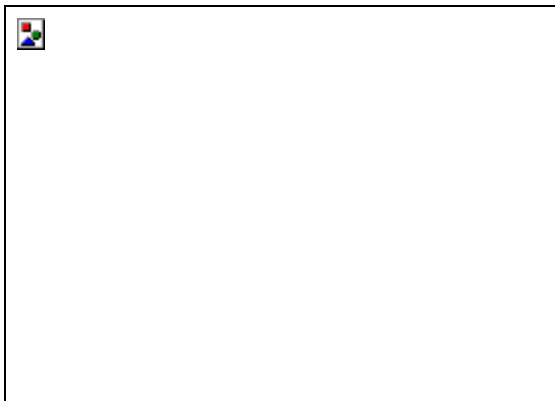


**Figure 4 :** *equipped operator.*



*Intervention site*

Suppose no picture displays on the PC's monitor. The operator will be guided by the system in order to determine the origin of the failure and oriented in order to check the graphic board inside the central unit. The following presentation illustrates an example of user/system interaction corresponding to this type of operations.



**Figure 5 :** *Internal components (power supply and electronic boards) displayed in a blended mode.*



**Figure 6 :** *Display of the graphic board.*

User : "Display internal components"

This query means : the user requires access to information database and rendering of this information without perceptual awareness (user action 2).

System : internal components of the central unit are displayed in the HMD of the user. Rendering is achieved in blended mode allowing to superimpose images of internal components to the real image of the central unit (fig. 5). These components are real-

time-registered to the real world. The operator can turn around the unit, and look inside as if it were transparent.

*User : "What to do when no picture displays on the monitor ?"*

This query is translated into the same actions as in the previous example (user action 2).

*System :* displays a textual window in the field of view of the operator. This window contains maintenance procedures to follow.

*User : "Display the graphic board"*

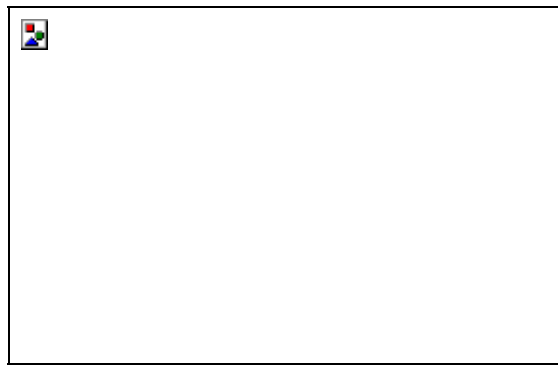
This query involves access to information and rendering of this information with perceptual awareness (the user wants to know the spatial location of a component). This is a user action 4.

*System :* the graphic board is displayed (fig. 6) (and possibly a blinking 3D arrow focuses on the desired board, not represented here).

The user will now perform measurements on the graphic port according to the maintenance procedure displayed earlier.



**Figure 7 :** wiring of the graphic port.



**Figure 8 :** 3D icons appear in space. They look as if they were tied to a fixed location in space. Such kind of mechanism can avoid an overflow display area.

*User : "Wiring of the graphic port"*

*System :* displays information panels in the field of view of the operator (fig. 7). Following these instructions, the operator will use his oscilloscope to visualise appropriate signals.

*User : "What can I do if the synchronisation signal looks wrong ?"*

*System :* Tells the user (through voice media) that he should activate a support decision system for a more complete diagnosis.

As the user doesn't need the oscilloscope tool any longer, he raises his hand in its direction and asks to iconify the corresponding window.

*User : "Iconify this window"*

The system combines perceptual awareness and rendering in order to properly modify visual aspect of the designated component (user action 3).

*System :* designated window is transformed into a 3D icon displayed in the working volume (fig. 8). Icon looks as if it were tied to a fixed location in space. If the operator turns his head, the icon leaves his field of view but still exists in the working volume. Later, the user can ask to see the information again or can move icons somewhere else in space. This mechanism is available for all 2D information.

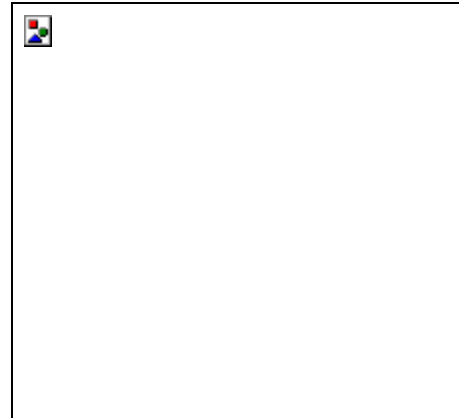
User : "Activation of my failure detection system"

System : The corresponding tool appears. The operator can now start a thorough testing session.

After tests have been performed the operator may want to retrieve the graphics board for testing it independently.

User : "Procedure for graphics board removal"

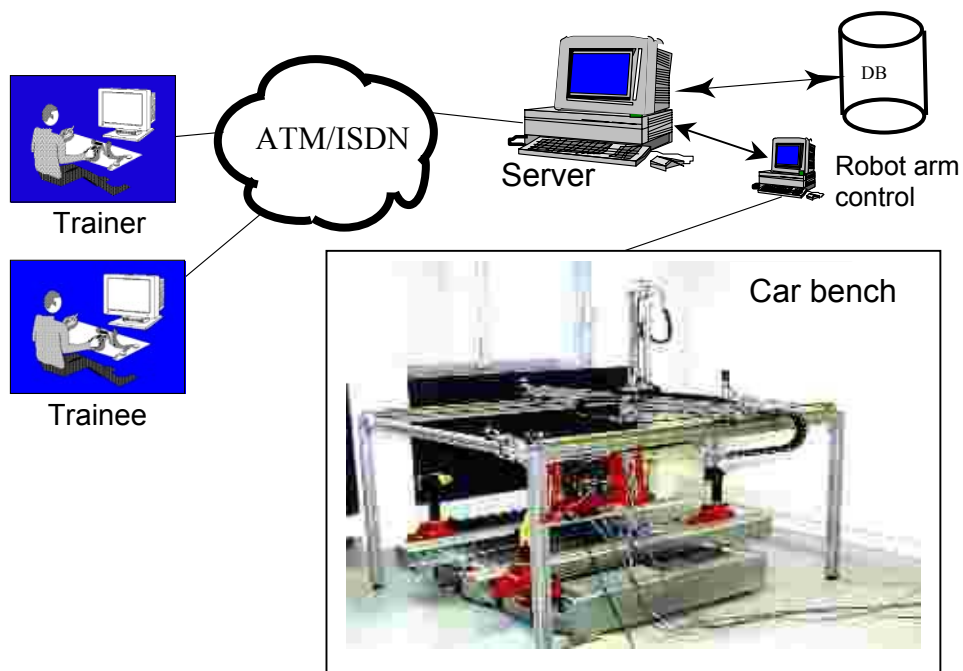
System : display relevant information for graphics board removal (fig. 9). A textual information bubble is registered to the board connector in the real video.



**Figure 9** : illustration of the graphics board removal procedure

### 3.2. Maestro system

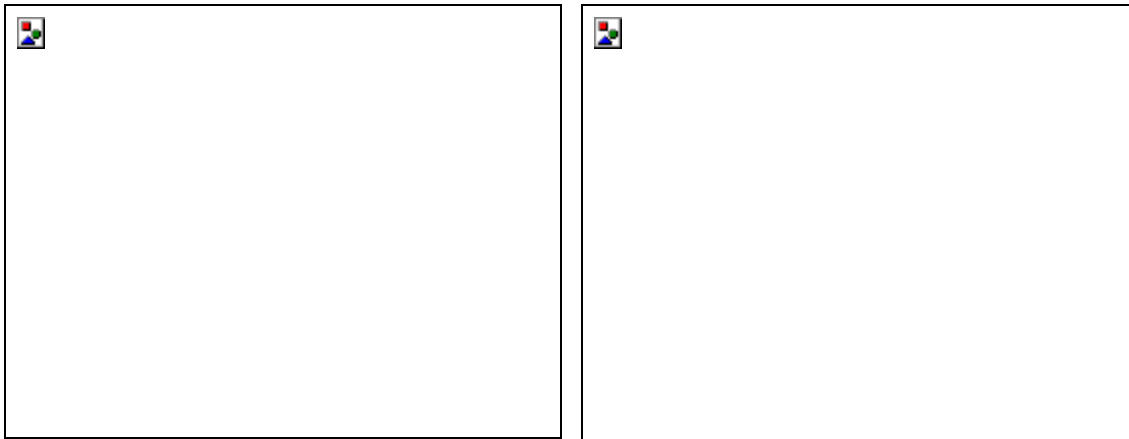
A second example of application is provided by the Maestro ACTS European Project (AC 233). Maestro system enables different users to share a synthetic environment where real and virtual information are mixed. This application allows remote instruction to maintenance and remote learning of maintenance tasks. One expert user is considered as the trainer while the other user is the trainee.



**Figure 10** : overview of the Maestro system.

The trainee is a car repairer who needs some help regarding the use of an equipment dedicated to car maintenance (such equipment is called a car bench). The trainer (the producer of the car bench) controls a remote robot arm (equipped with a video camera) in order to teach the trainee basic operations regarding the car bench (see fig. 10).

Like in the previous application users can query information databases through a speech based command system and see this information superimposed onto real objects (seen from the camera placed on the robot arm) as illustrated in fig. 11.



*“what are the different jig types”*

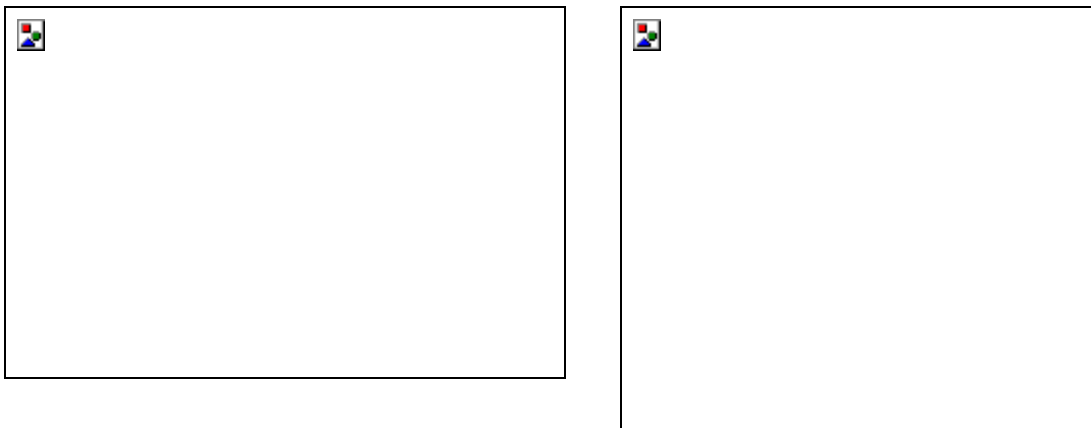
*“what are the different kind of beams”*

**Figure 11** : illustration of queries and answers from Maestro system.

This application is different from the previous one. Indeed, at least, two users share the same environment. Users do not directly control the virtual scene (*e.g.* with head movements) but act through a joystick which controls the robot arm. The application is distributed. This implies a client-server approach and a communication network. However, the overall architecture of the system is governed by the model presented above.

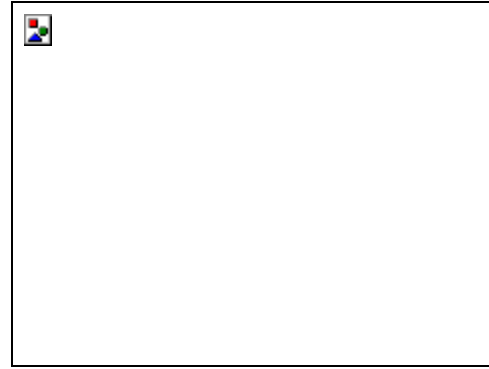
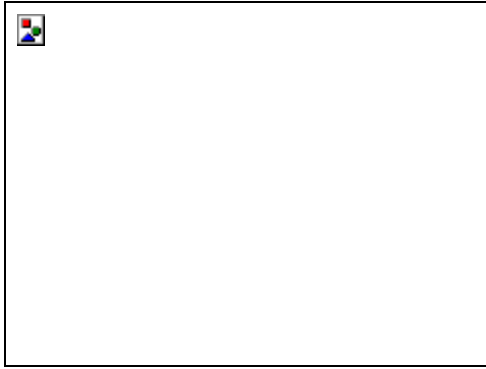
### 3.3. Mine clearance instruction

This application illustrates the use of an IDE in the field of defense and humanitarian operations. We have focused our work on a mine clearance instruction system dedicated to non governmental organizations. The system allows to guide the user in basic mine manipulation procedures (mine model discovery, manipulation and neutralization). Like in Virdoc, the user is equipped with a HMD, data glove and uses a speech control to communicate with the system. The user has the possibility to access general data concerning the type of mine he wants to work with, as illustrated in fig. 12.

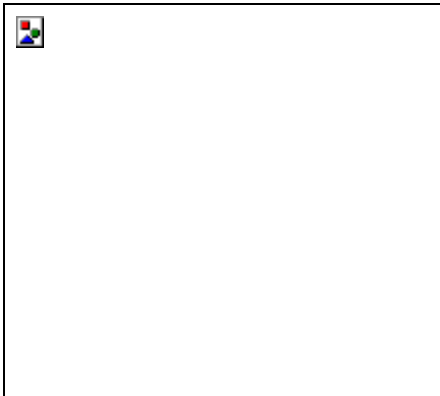


**Figure 12** : access to general information concerning the selected mine.

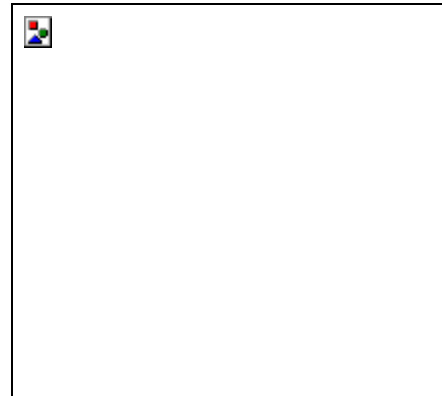
Another part of the scenario consists in illustrating how mine explosion is triggered. During the training session, the user is shown how to hold the mine in order that it remains all the time at the safe angle, as illustrated in fig.13. For this purpose, a spherical icon is superimposed on the video image of the mine, indicating its angle. As long as the angle remains in the safe area (the mine risk not to explode) the icon remains blue. When the angle attains the limit value the icon turns red thus indicating the mine would have blown if loaded.



**Figure 13 :** *mine explosion angle.*



**Figure 14 :** de-arming the mine.



**Figure 15 :** taking off the batteries.

Other parts of the scenario concern more technical manipulations like de-arming the mine or taking off the batteries of the detonator as illustrated in fig. 14 & 15. For this purpose, short videos demonstrating the appropriate procedure are displayed on bottom right corner of the main window.

This training scenario provides an example of a user-friendly system for mine clearance instruction, that could be used by laymen all over the world. Here again, the implementation follows the EDI model presented above.

Through these examples we can see that very different IDE applications can be designed based on the same generic model described before. These examples also reveal the importance of the operation scenario which makes the difference between toy problems and real applications. Design and validation of operation scenario are therefore critical tasks which should include application end-users.

#### **4. Conclusion**

We propose a model prescribing how to design an IDE. This model encompasses user/system interaction, system architecture and operation scenario. This presentation illustrates the basic components of IDE architecture (management of user requests, information storage, perception of the real environment, rendering of virtual information into real environment) and relations between these components. The importance of the operation scenario has been highlighted and a method for designing it proposed.

Based on this model, three illustrations have been presented. They provide examples of pre-operational applications in maintenance, tele-operation and tele-teaching systems and show the generic nature of the proposed model.

## 5. Bibliography

Chinnock, C., D. Calkins, et al. (1996). Hands-Free Mobile Computing : A new paradigm. L'Interface des mondes réels et virtuels, Montpellier, France.

Feiner, S., B. MacIntyre, et al. (1993). "Knowledge-based augmented reality." Communication of the ACM **36**(7): 53–62.

Takez, S., V. Conan, et al. (1997). The Virtually Documented Environments : A new Interface Paradigm for a Task-Oriented Access to Information. Computer Graphics, Budapest, Hungary, Addison Wesley.

Maes, P. (1994). "Agents that reduce work and information overload." Communication of the ACM **37**(7): 30-40.

Nagao, K. and J. Rekimoto (1995). Ubiquitous talker : spoken language interaction with real world objects. 14th International Joint Conference on Artificial Intelligence (IJCAI-95), (??).

Nagao, K. and J. Rekimoto (1996). Agent augmented reality : a software agent meets the real world. Second International Conference on Multiagent Systems (ICMAS-96).

Rekimoto, J. (1996). Transvision : A hand-Held Augmented Reality System for Collaborative Design. Virtual Systems and Multimedia (VSMM'96), (??).

Starner, T., S. Mann, et al. (1997). "Augmented Reality Through Wearable Computing." Presence **6**(4).

State, A., M. A. Livingstone, et al. (1996). Technologies for augmented reality systems : realizing ultrasound-guided needle biopsies. Siggraph'96, New-Orleans, USA.

Sutherland, I. (1968). A head-mounted three dimensional display. FJCC.

Wood, K. R., T. Richardson, et al. (1997). "Global Teleporting with Java : Toward Ubiquitous Personalized Computing." Computer : Innovative technology for computer professionals(Février): 53-59.

(1998). <http://www.intervisionsystems.com/>, Intervisions.

Rockwell (1998). <http://www.rockwell.com/>, Rockwell.

ViA (1998). <http://www.flexipc.com/>, ViA.

Xybernaut (1998). <http://www.xybernaut.com/>, Xybernaut Corporation.